Exhibit 5

Optimize network data usage

Over the life of a smartphone, the cost of a cellular data plan can easily exceed the cost of the device itself. On Android 7.0 (API level 24) and higher, users can enable Data Saver on a device-wide basis in order to optimize their device's data usage, and use less data. This ability is especially useful when roaming, near the end of the billing cycle, or for a small prepaid data pack.

When a user enables Data Saver in **Settings** and the device is on a metered network, the system blocks background data usage and signals apps to use less data in the foreground wherever possible. Users can allow specific apps to use background metered data usage even when Data Saver is turned on.

Android 7.0 (API level 24) extends the `ConnectivityManager` (/reference/android/net/ConnectivityManager) API to provide apps with a way to retrieve the user's Data Saver preferences (#status) and monitor preference changes (#monitor-changes). It is considered good practice for apps to check whether the user has enabled Data Saver and make an effort to limit foreground and background data usage.

Check data saver preferences

On Android 7.0 (API level 24) and higher, apps can use the `ConnectivityManager` (/reference/android/net/ConnectivityManager) API to determine what data usage restrictions are being applied. The `getRestrictBackgroundStatus()` (/reference/android/net/ConnectivityManager#getRestrictBackgroundStatus()) method returns one of the following values:

`RESTRICT_BACKGROUND_STATUS_DISABLED`

> Data Saver is disabled.

`RESTRICT_BACKGROUND_STATUS_ENABLED`

> The user has enabled Data Saver for this app. Apps should make an effort to limit data usage in the foreground and gracefully handle restrictions to background data usage.

`RESTRICT_BACKGROUND_STATUS_WHITELISTED`

> The user has enabled Data Saver but the app is allowed to bypass it. Apps should still make an effort to limit foreground and background data usage.

Limit data usage whenever the device is connected to a metered network, even if Data Saver is disabled or the app is allowed to bypass it. The following sample code uses `ConnectivityManager.isActiveNetworkMetered()` (/reference/android/net/ConnectivityManager#isActiveNetworkMetered()) and `ConnectivityManager.getRestrictBackgroundStatus()` to determine how much data the app should use:

Kotlin Java (#java)
(#kotlin)

```
(getSystemService(Context.CONNECTIVITY_SERVICE) as ConnectivityManager).apply {
    // Checks if the device is on a metered network
    if (isActiveNetworkMetered) {
        // Checks user's Data Saver settings.
        when (restrictBackgroundStatus) {
```

```
            RESTRICT_BACKGROUND_STATUS_ENABLED -> {
                // Background data usage is blocked for this app. Wherever possible,
                // the app should also use less data in the foreground.
            }
            RESTRICT_BACKGROUND_STATUS_WHITELISTED -> {
                // The app is allowed to bypass Data Saver. Nevertheless, wherever possible,
                // the app should use less data in the foreground and background.
            }
            RESTRICT_BACKGROUND_STATUS_DISABLED -> {
                // Data Saver is disabled. Since the device is connected to a
                // metered network, the app should use less data wherever possible.
            }
        }
    } else {
        // The device is not on a metered network.
        // Use data as required to perform syncs, downloads, and updates.
    }
}
```

Note: This behavior is different on Android TV. Instead of blocking background usage, Android TV only throttles it. When in the foreground, applications are limited to 800 Kbps, and when in the background, applications are limited to 10 Kbps. Use **`ConnectivityManager.isActiveNetworkMetered()`** to detect when to limit data usage on TV.

Request data restriction permissions

If your app needs to use data in the background, it can request data restriction permissions by sending a `Settings.ACTION_IGNORE_BACKGROUND_DATA_RESTRICTIONS_SETTINGS` intent containing a URI of your app's package name: for example, `package:MY_APP_ID`.

Sending the intent and URI launches the **Settings** app and displays data usage settings for your app. The user can then decide whether to enable background data for your app. Before you send this intent, it is good practice to first ask the user if they want to launch the **Settings** app for the purpose of enabling background data usage.

Monitor changes to data saver preferences

Apps can monitor changes to Data Saver preferences by creating a `BroadcastReceiver` (/reference/android/content/BroadcastReceiver) to listen for `ConnectivityManager.ACTION_RESTRICT_BACKGROUND_CHANGED` and dynamically registering the receiver with `Context.registerReceiver()` (/reference/android/content/Context#registerReceiver(android.content.BroadcastReceiver, android.content.IntentFilter)). When an app receives this broadcast, it should check if the new Data Saver preferences affect its permissions (#status) by calling `ConnectivityManager.getRestrictBackgroundStatus()`.

Note: The system only sends this broadcast to apps that dynamically register for them with **`Context.registerReceiver()`** (/reference/android/content/Context#registerReceiver(android.content.BroadcastReceiver, android.content.IntentFilter)). Apps that register to receive this broadcast in their manifest will not receive them.

Test with Android Debug Bridge commands

The Android Debug Bridge (ADB) (/tools/help/adb) provides a few commands that you can use to test your app in Data Saver conditions. You can check and configure network permissions or set wireless networks as metered to test your app on unmetered networks.

`$ adb shell dumpsys netpolicy`

Generates a report that includes the current global background network restriction setting, package UIDs that are currently allowed to bypass Data Saver, and the network permissions of other known packages.

`$ adb shell cmd netpolicy`

Displays a full list of Network Policy Manager (netpolicy) commands.

`$ adb shell cmd netpolicy set restrict-background <boolean>`

Enables or disables Data Saver mode when passing `true` or `false`, respectively.

`$ adb shell cmd netpolicy add restrict-background-whitelist <UID>`

Adds the specified package UID to the allowlist (`whitelist`) to allow background metered data usage.

`$ adb shell cmd netpolicy remove restrict-background-whitelist <UID>`

Removes the specified package UID from the allowlist (`whitelist`) to block background metered data usage while Data Saver is enabled.

`$ adb shell cmd netpolicy list wifi-networks`

Lists all wifi networks, displaying whether they're metered.

`$ adb shell cmd netpolicy set metered-network <WIFI_SSID> true`

Sets wifi with the specified SSID as metered, allowing you to simulate a metered network on an unmetered network.

Last updated 2024-01-03 UTC.